De nouvelles meilleures solutions pour le problème d'ordonnancement No-Wait Flowshop

Lucien Mousin, Marie-Eléonore Kessaci, Clarisse Dhaenens

De nouvelles meilleures solutions pour le problème d'ordonnancement *No-Wait Flowshop*

Lucien Mousin[1], Marie-Eléonore Kessaci[1], Clarisse Dhaenens[1]

Université de Lille, INRIA, CNRS, UMR 9189 - CRIStAL, France
{lucien.mousin}@ed.univ-lille1.fr
{me.kessaci, clarisse.dhaenens}@univ-lille1.fr

Mots-clés : *ordonnancement de type flowshop, heuristique, métaheuristique*

1 Introduction

Le problème No-Wait Flowshop (NWFSP) est une variante du problème d'ordonnancement de type flowshop de permutation où aucun temps d'attente n'est autorisé entre l'exécution de chaque tâche sur les machines successives. Ainsi l'exécution d'une tâche est exactement le temps nécessaire pour effectuer chaque tâche par chaque machine contrairement au problème classique. Cette particularité lui confère des propriétés et une structure intéressantes qui peuvent être utilisées dans des algorithmes de résolution tels que les heuristiques ou les métaheuristiques. Partant de cette observation, nous proposons une méthode rapide pour construire des solutions initiales meilleure que celles construites par les heuristiques constructives de la littérature. Cette méthode d'initialisation sera ensuite utilisée comme point de départ à une nouvelle métaheuristique nous permettant d'obtenir de nouvelles meilleures solutions ayant des qualités non encore atteintes actuellement pour les instances de Taillard [3].

2 L'heuristique Iterated Best Insertion (IBI)

Les heuristiques constructives agissent de manière gloutonne en insérant à chaque itération, une des tâches non encore ordonnancée. Nous avons repris cette idée en ajoutant à la fin de chaque itération, une recherche locale pour obtenir pour chaque sous-problème traité un optimum local (voire global dans certain cas). La recherche locale choisie (amélioration itérative) est très basique et se base sur l'opérateur d'insertion. Elle consiste à se déplacer vers un voisin améliorant en utilisant l'opérateur d'insertion et s'arrête dès qu'une solution n'a pas de voisin améliorant.

Dans ce travail, le but est de trouver un ordonnancement qui minimise le *makespan* (noté C_{max}) défini comme le temps total d'exécution. Cette méthode a montré de très bonnes performances pour construire des solutions initiales rapidement pour les instances de Taillard.

3 Les Super-tâches : des sous-séquences pour réduire la combinatoire

Le NWFSP présente une structure particulière qui confère aux bonnes solutions des propriétés intéressantes à exploiter. En effet, en observant les solutions d'un problème, on remarque des similarités comme l'existence de couples de tâches qui sont toujours ensemble dans le même ordre dans les solutions de bonne qualité. Nous proposons d'utiliser chacune de ces *sous-séquences* de tâches comme une seule *super-tâche*. Ceci peut permettre de réduire la taille

du problème et donc la combinatoire associée à sa résolution. De même, la structure du paysage est modifiée et les métaheuristiques peuvent ainsi se comporter différemment.

L'algorithme Iterated Greedy (IG) [2] a démontré son efficacité pour résoudre le problème d'ordonnance de type flowshop classique en partie grâce à sa stratégie de déconstruction-construction. Celle-ci a été utilisée dans une recherche locale VNS [1] qui a permis d'obtenir la majorité des meilleures solutions connues pour ce problème. En effet, l'IG, même s'il est efficace, ne permet pas dans sa version originale d'obtenir de nouvelles meilleures solutions même si les solutions obtenues s'en approchent en qualité. C'est pour cette raison, que nous utilisons IG pour obtenir un ensemble S de bonnes solutions diversifiées pour pouvoir construire nos *super-tâches*.

Pour commencer, nous fixons un seuil α à 55% qui détecte les *super-tâches* présentes dans 55% des solutions de S. Le niveau assez bas de ce seuil nous permet d'obtenir beaucoup de *super-tâches* et d'abaisser fortement la taille du problème traité *ie.* le nombre de tâches à ordonnancer. Puis, nous utilisons IBI pour construire rapidement et de manière efficace une solution initiale. Puis, IG est appliqué un petit nombre d'itérations, et permet d'obtenir une solution de meilleure qualité. Le seuil α est ré-haussé à 75% pour ne conserver qu'une partie des *super-tâches*. Le nombre de tâches à ordonnancer augmente. La solution obtenue à l'étape précédente est reprise, et l'IG est ré-appliqué quelques itérations. Enfin, on revient au problème initial en considérant chacune des tâches initiales pour appliquer une dernière fois IG.

4 Résultats et conclusion

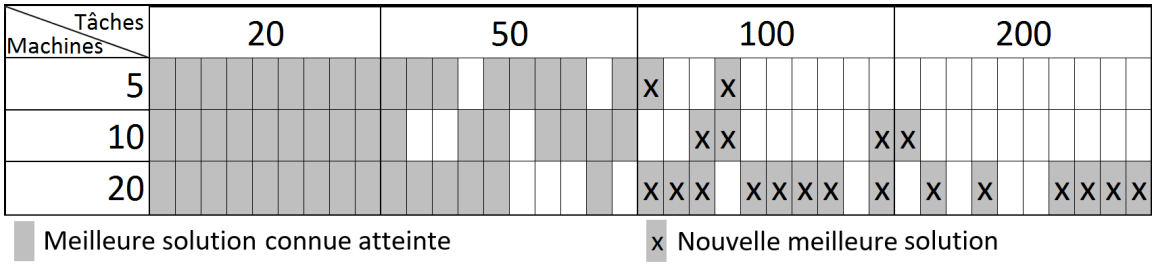

FIG. 1 – Résultats sur les instances de Taillard pour une exécution pour chaque instance de chaque taille.

La méthode décrite ci-dessus, nous permet d'obtenir de nouvelles meilleures solutions pour les instances de Taillard (voir Figure 1). Les *super-tâches* semblent être un bon moyen pour améliorer les performances de l'algorithme IG sur le problème du NWFSP.

Cette étude met en évidence l'intérêt d'extraire de la connaissance de solutions connues pour permettre d'atteindre de nouvelles zones de l'espace de recherche. En plus de réduire la combinatoire du problème, la fusion des tâches permet aux méthodes d'optimisation de trouver plus facilement de nouvelles solutions de très bonne qualité.

L'utilisation des sous-séquences semble donc prometteuse et il serait intéressant d'adapter ce principe sur d'autres problèmes de permutation afin de généraliser cette approche.

Références

[1] J-Y. Ding, S. Song, J. N.D. Gupta, R. Zhang, R. Chiong, and C. Wu. An improved iterated greedy algorithm with a tabu-based reconstruction strategy for the no-wait flowshop scheduling problem. *Applied Soft Computing*, 30 :604–613, 2015.

[2] R. Ruiz and T. Stützle. A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem. *EJOR*, 177(3) :2033–2049, 2007.

[3] E. Taillard. Benchmarks for basic scheduling problems. *EJOR*, 64(2) :278–285, 1993.